

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-108656

(43)Date of publication of application : 12.04.2002

(51)Int.Cl.

G06F 11/30

(21)Application number : 2000-298387

(71)Applicant : NEC CORP

(22)Date of filing : 29.09.2000

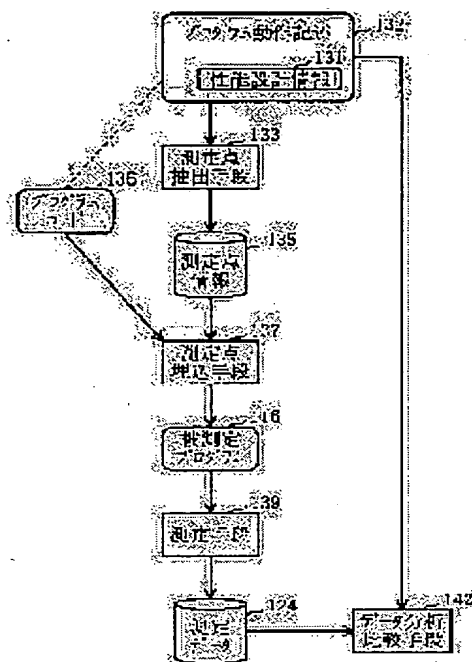
(72)Inventor : HORIKAWA TAKASHI

## (54) PROGRAM OPERATION VERIFYING SYSTEM

### (57)Abstract:

**PROBLEM TO BE SOLVED:** To provide a program operation verifying system in which a measuring point can be arranged flexibly on a program to be measured and the operation situation of the program to be measured can be easily verified when it is executed on an actual device.

**SOLUTION:** A measuring point extracting means 133 extracts measuring points from program operation description 132 including performance design information 131 and a measuring point embedding means 137 embeds the measuring points (a start point and an end point) into a program code to prepare a program 116 to be measured. When this program is executed, a measuring means 139 outputs the time and event at each measuring point to generate measurement data 124. Thus, it is possible to learn the lapsed time between the measuring points when the program is executed. Therefore, it is possible to learn the operation situation and to verify the operation. This system can also be applied even to a performance index (for example, a CPU consuming time and the number of times of access of a disk or a network) other than the lapsed time.



## LEGAL STATUS

[Date of request for examination]

21.08.2001

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2002-108656

(P2002-108656A)

(43) 公開日 平成14年4月12日 (2002.4.12)

(51) Int.Cl.<sup>7</sup>

G 0 6 F 11/30

識別記号

3 1 0

F I

G 0 6 F 11/30

テーマト\* (参考)

3 1 0 A 5 B 0 4 2

審査請求 有 請求項の数 9 O L (全 14 頁)

(21) 出願番号 特願2000-298387 (P2000-298387)

(22) 出願日 平成12年9月29日 (2000.9.29)

(71) 出願人 000004237

日本電気株式会社

東京都港区芝五丁目7番1号

(72) 発明者 堀川 隆

東京都港区芝五丁目7番1号 日本電気株式会社内

(74) 代理人 100083987

弁理士 山内 梅雄

Fターム (参考) 5B042 HH20 HH25 LA02 MA14 MA20

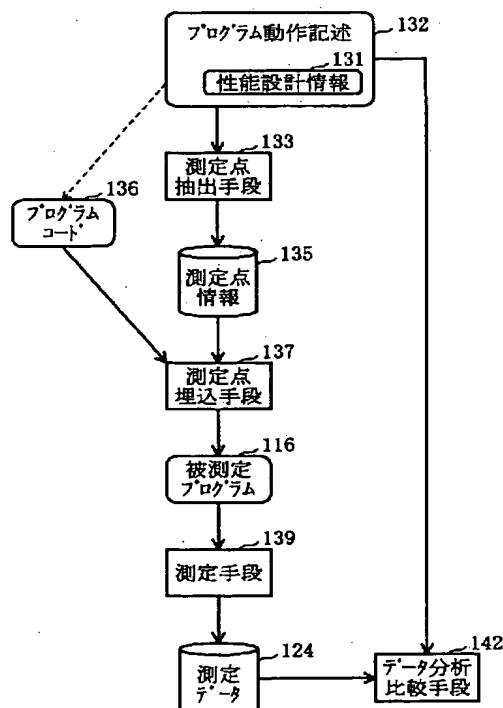
MB01 MB05 MB06 MC33

(54) 【発明の名称】 プログラム動作検証システム

(57) 【要約】 (修正有)

【課題】 被測定プログラム上に柔軟に測定点を配置することができ、被測定プログラムが実際の装置で実行されるとき動作状況を容易に検証することのできるプログラム動作検証システムを提供する。

【解決手段】 測定点抽出手段133は性能設計情報131を含んだプログラム動作記述132から測定点を抽出し、測定点埋込手段137によってプログラムコードに測定点(開始点および終了点)を埋め込んで被測定プログラム116を作成する。これを実行すると、測定手段139は各測定点でそれらの時刻とイベントを出力して測定データ124とする。測定データ124によってプログラム実行時の測定点間の経過時間が分かる。これにより、動作状況を知り動作検証を行うことができる。経過時間以外の性能指標(たとえばCPU消費時間、ディスクやネットワークのアクセス回数)にも本発明を適用できる。



## 【特許請求の範囲】

【請求項 1】 プログラムのある特定範囲の処理に要する経過時間等の所定の性能指標を調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点および測定終了のための測定終了点を読み出す読出手段と、この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、

この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、

このプログラム実行手段の被測定プログラムの実行の際に前記測定開始点および測定終了点の埋め込み箇所の処理が実行されるときに性能指標をそれぞれ測定する性能指標測定手段と、

この時刻測定手段によって測定された測定開始点と測定終了点のそれぞれの性能指標の差から前記被測定プログラムのこれら測定開始点から測定終了点に至る性能指標値を算出する性能指標値算出手段と、

この性能指標値算出手段によって算出した性能指標値を前記被測定プログラムの対応する箇所と共に表示する表示手段とを具備することを特徴とするプログラム動作検証システム。

【請求項 2】 プログラムのある特定範囲の処理に要する経過時間等の所定の性能指標がその処理について予め定めた設計値の範囲内に収まっているかどうかを少なくとも調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点と測定終了のための測定終了点および前記設計値を読み出す読出手段と、

この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、

この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、

このプログラム実行手段の被測定プログラムの実行の際に前記測定開始点および測定終了点の埋め込み箇所の処理が実行されるときに性能指標をそれぞれ測定する性能指標測定手段と、

この性能指標測定手段によって測定された測定開始点と測定終了点のそれぞれの性能指標の差から前記被測定プログラムのこれら測定開始点から測定終了点に至る性能指標値を算出する性能指標値算出手段と、

この性能指標値算出手段によって算出した性能指標値を前記被測定プログラムの対応する箇所ならびに前記設計値と共に表示する表示手段とを具備することを特徴とするプログラム動作検証システム。

【請求項 3】 プログラムのある特定範囲の処理に要す

る経過時間等の所定の性能指標がその処理について予め定めた設計値の範囲内に収まっているかどうかを少なくとも調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点と測定終了のための測定終了点および前記設計値を読み出す読出手段と、

この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、

10 この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、

このプログラム実行手段の被測定プログラムの実行の際に前記測定開始点および測定終了点の埋め込み箇所の処理が実行されるときに性能指標をそれぞれ測定する性能指標測定手段と、

この性能指標測定手段によって測定された測定開始点と測定終了点の性能指標の差から前記被測定プログラムのこれら測定開始点から測定終了点に至るまでのプログラムの実行に要する性能指標値を算出する性能指標値算出手段と、

20 この性能指標値算出手段によって算出した性能指標値が前記設計値を満足するか否かを判別する判別手段と、この判別手段によって判別した結果を前記被測定プログラムの対応する箇所と共に表示する表示手段とを具備することを特徴とするプログラム動作検証システム。

【請求項 4】 性能指標は、プログラムのある特定範囲の処理に要する経過時間の他に、CPU消費時間、ディスクやネットワークのアクセス回数、入出力データのサイズ等の指標であることを特徴とする請求項 1～請求項 3 記載のプログラム動作検証システム。

【請求項 5】 プログラムのある特定範囲の処理に要する経過時間がその処理について予め定めた設計値の範囲内に収まっているかどうかを少なくとも調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点と測定終了のための測定終了点および前記設計値を読み出す読出手段と、

この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、

この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、

このプログラム実行手段の被測定プログラムの実行の際に前記測定開始点および測定終了点の埋め込み箇所の処理が実行されるときに時刻をそれぞれ測定する時刻測定手段と、

50 この時刻測定手段によって測定された測定開始点と測定終了点の時刻の差から前記被測定プログラムのこれら測

## 3

定開始点から測定終了点に至るまでのプログラムの実行に要する経過時間を算出する経過時間算出手段と、この経過時間算出手段によって算出した経過時間が前記設計値を満足するか否かを判別する判別手段と、この判別手段によって判別した結果を前記被測定プログラムの対応する箇所と共に表示する表示手段とを具備することを特徴とするプログラム動作検証システム。

【請求項6】 前記表示手段は経過時間算出手段によって算出した実際の経過時間も前記被測定プログラムの対応する箇所と共に表示することを特徴とする請求項5記載のプログラム動作検証システム。

【請求項7】 前記所定の装置は複数の装置によって構成されており、前記プログラムのある特定範囲の処理とは被測定プログラムのうち特定の装置に割り当てられたコンポーネントの処理であることを特徴とする請求項1～請求項3または請求項5記載のプログラム動作検証システム。

【請求項8】 前記所定の装置は通信ケーブルによって接続された複数の装置によって構成されており、それぞれの装置に対して前記被測定プログラムのコンポーネントが割り振られており、前記表示手段はそれぞれのコンポーネントについての表示を一箇所で行うことを特徴とする請求項1～請求項3または請求項5記載のプログラム動作検証システム。

【請求項9】 前記所定の装置は通信ケーブルによって接続された複数の装置によって構成されており、それぞれの装置に対して前記被測定プログラムのコンポーネントが割り振られており、前記表示手段はそれぞれの装置単位で該当するコンポーネントごとの表示を行うことを特徴とする請求項1～請求項3または請求項5記載のプログラム動作検証システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明はプログラムの動作を検証するためのプログラム動作検証システムに係わり、特にプログラムの開発時に実際の計算機等でその動作を検証するためのプログラム動作検証システムに関する。

【0002】

【従来の技術】ソフトウェア工学の進展によって、比較的大きなソフトウェアをその構成部品としてのコンポーネントを組み合わせるというコンポーネント指向開発が普及しつつある。コンポーネントとは、ある機能やサービスを提供するための独立して交換可能なソフトウェアの構成単位をいう。

【0003】このような開発手法をサポートする技術の1つにUML (Unified Modeling Language) がある。UMLはソフトウェアの仕様や、図示、体系化、文書化の方法を規定したシステム設計用のモデリング言語である。UMLを利用すると、高品質なシステム構築を可能にするだけでなく、リアルタイムシステムや分散システ

## 4

ム等のさまざまなシステム設計を簡素化することができる。UMLはOMG (Object Management Group, Inc.) が標準化を行っている。このUMLで規定されている記述方法の1つであるシーケンス図は、作成したソフトウェアがコンピュータ等の装置で実際に処理されるとき動作を、前記したコンポーネントあるいはそれを構成するオブジェクトの間で行われる通信といった相互作用を時間軸に沿って記述するものである。

【0004】このシーケンス図は、ソフトウェアの動作を時間軸で表わしているため、この図の中に時間に関する性能情報を含めることが容易にできる。たとえばあるコンポーネントの処理は3秒間を要したといった情報を含めてソフトウェアの動作を表現することができる。実際に、UMLの第1.3版の仕様書 (OMG Unified Modeling Language Specification, <http://cgi.omg.org/cgi-bin/doc?formal/00-03-0.1.pdf>) の3-96ページに示されているシーケンス図には、プログラムの実行に関する性能条件も一緒に記述している。このような性能条件の集まりを本明細書では性能設計情報と呼ぶことにする。

【0005】一方、計算機の従来の性能測定は、マイクロソフト社のOS (オペレーションシステム) としてのウィンドウズで使用されているパフォーマンスモニタ (performance monitor) やユニックス (UNIX (登録商標)) におけるSAR (システム動作情報表示機能) のようにCPU (中央処理装置) やディスクといった計算機資源の使用率の計測が一般的である。

【0006】これら従来から使用されたツールは、一定の周期で測定対象のシステムの状態を調べて計算機資源の使用率を見積もるサンプリング手法をベースとしている。また、計算機資源の使用開始や終了といったイベント発生時に測定操作を行うことで、計算機資源の使用時間を測定するようにしたイベントドリブン方式のツールも存在している。

【0007】これらはいずれも、測定期間中の計算機資源の使用率 (あるいは使用時間) を測定するものである。すなわちこれらの性能測定は、性能設計情報としてのシーケンス図で記述された性能条件に直接対応するものではない。したがって、たとえばあるオブジェクトが処理要求メッセージを受信してからリプライを返すまでの処理に要した計算機資源量を測定するといった、性能条件で引用されているプログラム実行途中の2点の処理を測定することはできない。このため、前記したようにソフトウェアを作成するときその構成部品としてのコンポーネントごとの性能を評価するといったことができない。

【0008】ところで、特開平11-96046号公報ではプログラムが動作する計算機システムからプログラムの稼動状況を取得する際に、予め該当するプログラム、収集タイミング、収集するデータの項目等を含んだ

収集情報選択定義情報を外部から定義情報格納手段に格納するようにしている。この技術では、熟練したソフトウェア開発者が収集情報選択定義情報を定義情報格納手段に格納することでこれらの者のノウハウを活かして誤った情報を収集したり不十分な情報を収集するといった不都合を解消している。しかしながら、このような技術で活用される定義情報格納手段では該当するプログラムを指定するのは人手に頼ることになる。また、熟練者のノウハウを活用するということは定義情報格納手段に格納する情報の入力を行う者は熟練者であることを必要とするという問題がある。

【0009】一方、特開平9-18303号公報では、現在使用されているプログラム言語よりも上位レベルの記述を利用して、分散アプリケーションを自動構築する手法の分野で、上位レベル記述から測定点を抽出して、プローブを自動的に埋め込む処理を開示している。また、特許第2924436号公報では、性能測定対象のプロセスに遠隔手続き呼び出しがあることを前もって検出して性能の測定を行うシステムを開示している。

【0010】更に特開平8-314767号公報では、OS機能として用意されているイベント収集機能を使用して計算機の性能を測定する技術を開示している。すなわち、この技術ではタスクの実行中にOSによって収集されるイベント情報を時系列的に格納して、全タスクの実行後に、これらのイベント情報を順次読み出して、タスクごとにデータベースのリソースや入力操作に関するデータベース関連のイベント情報を抽出し、データベースの操作性に関する複数の評価パラメータを集計するようにしている。

【0011】

【発明が解決しようとする課題】しかしながら、これらの技術は先の特開平11-96046号公報とも共通するが、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行っていたり、メッセージの送受信点といったように予め決まったルールを使用したり、OS自体が用意している機能を利用するようになっている。

【0012】このため、手作業で行う場合には得られた測定結果に人為的な影響が作用してしまい、測定結果が妥当であるかどうかの判断が困難であるという問題があった。また、測定の実施者が特別な知識や経験を有する者に限定されるだけでなく、作業に時間を要するといった問題があった。更に、予め決まったルールや機能を使用する場合には、手作業で行う場合の問題がなくなる一方で、プログラム上に測定点を設置することができる位置が、これらのルールや機能を適用できる箇所に限られることになる。したがって、測定を柔軟に行うことができないという問題があった。

【0013】このため、性能設計情報を含むプログラム動作記述をもとに開発されたプログラムがこれらの性能

設計情報で示されている性能条件（たとえば、プログラム実行中における任意の点を参照する）を満足しているかどうかを調べるための測定を行うものとする、従来ではソフトウェアの開発者が手作業で性能設計情報に対応する測定点を抽出していた。そして、これらの測定点を用いて、プログラムの実行に要した時間が適切な時間範囲であったかといった性能測定を実施する必要があった。このために、前記した手作業が原因で測定の実施に手間がかかるといった問題があった。

10 【0014】またこのような従来の手法によると、測定点の抽出操作を個々のプログラムごとに個別に行う必要があった。このように、性能設計情報に対応させた測定を行う測定手法を汎用化された測定ツールという形で提供するまでに至っておらず、これが測定の実施に時間がかかる一因ともなっていた。また、予め決まったルールや機能を使用する場合には、測定点が一方的に定められることになり、必要な測定点を任意に抽出することができない。したがって、性能設計を行う際に性能条件を自由に記述することができないという問題があった。

20 【0015】そこで本発明の目的は、被測定プログラム上に柔軟に測定点を配置することができ、被測定プログラムが実際の装置で実行されるとき動作状況を容易に検証することのできるプログラム動作検証システムを提供することにある。

【0016】

【課題を解決するための手段】請求項1記載の発明では、(イ)プログラムのある特定範囲の処理に要する経過時間等の所定の性能指標を調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点および測定終了のための測定終了点を読み出す読出手段と、

30 (ロ)この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、(ハ)この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、(ニ)このプログラム実行手段の被測定プログラムの実行の際に測定開始点および測定終了点の埋め込み箇所の処理が実行されるとき性能指標をそれぞれ測定する性能指標測定手段と、(ホ)この時刻測定手段によって測定された測定開始点と測定終了点のそれぞれの性能指標の差から被測定プログラムのこれら測定開始点から測定終了点に至る性能指標値を算出する性能指標値算出手段と、(ヘ)この性能指標値算出手段によって算出した性能指標値を被測定プログラムの対応する箇所と共に表示する表示手段とをプログラム動作検証システムに具備させる。

50 【0017】すなわち請求項1記載の発明では、読出手段によって性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開

始点と測定終了点および設計値を読み出して、被測定プログラム作成手段で測定開始点と測定終了点をプログラムの該当箇所に埋め込んで被測定プログラムを作成するようにしている。そして、時刻測定手段によってプログラム実行手段の被測定プログラムの実行の際に測定開始点および測定終了点の埋め込み箇所の処理が実行されるときに性能指標をそれぞれ測定し、それぞれの性能指標の差から被測定プログラムのこれら測定開始点から測定終了点に至る性能指標値を算出するようにしている。算出した性能指標値は判別手段を用いて設計値を満足する  
10 可否かを判別し、判別結果が被測定プログラムの対応する箇所に表示される。このように請求項 1 記載の発明では、プログラム動作記述の性能設計情報に記述した内容を基にして測定開始点と測定終了点を抽出するので、たとえばプログラマが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所で測定を開始したり終了させることができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージの送受信点と  
20 いったように予め決まったルールを使用したり、OS 自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での性能指標を求め、これらの差から性能指標値を算出しこれを設計値と比較した判別結果を被測定プログラムの対応する箇所に対応付けて表示するので、チェックするコンポーネントが多いような場合にもそれぞれの性能指標値が設計値を満足しているかどうかを短時間にチェックすることができる。

【0018】請求項 2 記載の発明では、(イ) プログラムのある特定範囲の処理に要する経過時間等の所定の性能指標がその処理について予め定めた設計値の範囲内に収まっているかどうかを少なくとも調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点と測定終了のための測定終了点および設計値を読み出す読  
30 出手段と、(ロ) この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、(ハ) この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、(ニ) このプログラム実行手段の被測定プログラムの実行の際に測定開始点および測定終了点の埋め込み箇所の処理が実行されるときに性能指標をそれぞれ測定する性能指標測定手段と、(ホ) この性能指標測定手段によって測定された測定開始点と測定終了点のそれぞれの性能指標の差から被測定プログラ  
40 ムのこれら測定開始点から測定終了点に至る性能指標値を算出する性能指標値算出手段と、(ヘ) この性能指

標値算出手段によって算出した性能指標値を被測定プログラムの対応する箇所に並びに設計値と共に表示する表示手段とをプログラム動作検証システムに具備させる。

【0019】すなわち、請求項 2 記載の発明では、プログラム動作記述の性能設計情報に記述した内容を基にして測定開始点と測定終了点を抽出するので、たとえばプログラマが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所で測定を開始したり終了させる  
10 ことができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージの送受信点とといったように予め決まったルールを使用したり、OS 自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での性能指標を求め、これらの差から性能指標値を算出して表示するので、性能指標値を被測定プログラ  
20 ムの対応する箇所に対応付けて簡単にチェックすることができる。更に本発明では、算出した性能指標値を設計値と共に被測定プログラムの対応する箇所に表示するので、算出した性能指標値が設計値を満足しているかを簡単にチェックすることができる。

【0020】請求項 3 記載の発明では、(イ) プログラムのある特定範囲の処理に要する経過時間等の所定の性能指標がその処理について予め定めた設計値の範囲内に収まっているかどうかを少なくとも調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点と測定終了のための測定終了点および設計値を読み出す読  
30 出手段と、(ロ) この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、(ハ) この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、(ニ) このプログラム実行手段の被測定プログラムの実行の際に測定開始点および測定終了点の埋め込み箇所の処理が実行されるときに性能指標をそれぞれ測定する性能指標測定手段と、(ホ) この性能指標測定手段によって測定された測定開始点と測定終了点の性能指標の差から被測定プログラ  
40 ムのこれら測定開始点から測定終了点に至るまでのプログラムの実行に要する性能指標値を算出する性能指標値算出手段と、(ヘ) この性能指標値算出手段によって算出した性能指標値が設計値を満足するかどうかを判別する判別手段と、(ト) この判別手段によって判別した結果を被測定プログラムの対応する箇所と共に表示する表示手段とをプログラム動作検証システムに具備させる。

【0021】すなわち請求項 3 記載の発明では、プログラム動作記述の性能設計情報に記述した内容を基にして  
50

測定開始点と測定終了点を抽出するので、たとえばプログラムが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所で測定を開始したり終了させることができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージの送受信点といったように予め決まったルールを使用したり、OS自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での性能指標を求め、これらの差から性能指標値を算出しこれを設計値と比較した判別結果を被測定プログラムの対応する箇所に対応付けて表示するので、チェックするコンポーネントが多いような場合にもそれぞれの性能指標値が設計値を満足しているかどうかを短時間にチェックすることができる。

【0022】請求項4記載の発明では、性能指標は、たとえばプログラムのある特定範囲の処理に要する経過時間の他に、CPU消費時間、ディスクやネットワークのアクセス回数、入出力データのサイズ等の指標であることを示している。

【0023】請求項5記載の発明では、(イ)プログラムのある特定範囲の処理に要する経過時間がその処理について予め定めた設計値の範囲内に収まっているかどうかを少なくとも調べる性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始のための測定開始点と測定終了のための測定終了点および設計値を読み出す読出手段と、(ロ)この読出手段で読み出した測定開始点および測定終了点を該当する箇所にそれぞれ埋め込んだ被測定プログラムを作成する被測定プログラム作成手段と、(ハ)この被測定プログラム作成手段によって作成された被測定プログラムを所定の装置に組み込んで実行するプログラム実行手段と、(ニ)このプログラム実行手段の被測定プログラムの実行の際に測定開始点および測定終了点の埋め込み箇所の処理が実行されるとき時刻をそれぞれ測定する時刻測定手段と、(ホ)この時刻測定手段によって測定された測定開始点と測定終了点の時刻の差から被測定プログラムのこれら測定開始点から測定終了点に至るまでのプログラムの実行に要する経過時間を算出する経過時間算出手段と、(ヘ)この経過時間算出手段によって算出した経過時間が設計値を満足するか否かを判別する判別手段と、(ト)この判別手段によって判別した結果を被測定プログラムの対応する箇所と共に表示する表示手段とをプログラム動作検証システムに具備させる。

【0024】すなわち請求項5記載の発明では、読出手段によって性能設計情報を含んだプログラム動作記述から検証の対象となる被測定プログラムに埋め込む測定開始点と測定終了点および設計値を読み出して、被測定プ

ログラム作成手段で測定開始点と測定終了点をプログラムの該当箇所に埋め込んで被測定プログラムを作成するようにしている。そして、時刻測定手段によってプログラム実行手段の被測定プログラムの実行の際に測定開始点および測定終了点の埋め込み箇所の処理が実行されるとき時刻をそれぞれ測定し、被測定プログラムのこれら測定開始点から測定終了点に至るまでのプログラムの実行に要する経過時間を経過時間算出手段によって算出するようにしている。算出した経過時間は判別手段を用いて設計値を満足するか否かを判別し、判別結果が被測定プログラムの対応する箇所に表示される。

【0025】このように請求項5記載の発明では、プログラム動作記述の性能設計情報に記述した内容に基づいて測定開始点と測定終了点を抽出するので、たとえばプログラムが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所で測定を開始したり終了させることができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージの送受信点といったように予め決まったルールを使用したり、OS自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での処理時刻を求め、これらの差から経過時間を算出しこれを設計値と比較した判別結果を被測定プログラムの対応する箇所に対応付けて表示するので、チェックするコンポーネントが多いような場合にもそれぞれの所要時間が設計値を満足しているかどうかを短時間にチェックすることができる。

【0026】請求項6記載の発明では、請求項5記載のプログラム動作検証システムで、表示手段は経過時間算出手段によって算出した実際の経過時間も被測定プログラムの対応する箇所と共に表示することを特徴としている。

【0027】すなわち請求項6記載の発明では、請求項3記載の発明による表示内容に加えて経過時間算出手段によって算出した実際の経過時間も表示するようにしているので、この経過時間を設計値と対比することでプログラムの実行について何らかの不具合があったときにその原因を大まかに類推することができる。

【0028】請求項7記載の発明では、請求項1～請求項3または請求項5記載のプログラム動作検証システムで、前記した所定の装置は複数の装置によって構成されており、プログラムのある特定範囲の処理とは被測定プログラムのうち特定の装置に割り当てられたコンポーネントの処理であることを特徴としている。

【0029】すなわち請求項7記載の発明では、1つの被測定プログラムが複数の装置によってコンポーネント単位で実行される例を示している。このようなコンポー

ネットごとの経過時間を表示したりそれぞれの設計値と比較することがコンポーネント単位で測定開始点と測定終了点をプログラム動作記述から抽出し被測定プログラムに埋め込むことで実現可能になる。

【0030】請求項8記載の発明では、請求項1～請求項3または請求項5記載のプログラム動作検証システムで、前記した所定の装置は通信ケーブルによって接続された複数の装置によって構成されており、それぞれの装置に対して被測定プログラムのコンポーネントが割り振られており、表示手段はそれぞれのコンポーネントについての表示を一箇所で集中的に行うことを特徴としている。

【0031】すなわち請求項8記載の発明では、1つの被測定プログラムが複数の装置によって実行されるとき、それぞれの装置におけるコンポーネントごとの検証結果は1つの装置で集中的に表示される例を示しており、これにより被測定プログラムの動作全体についての集中的な検証が可能になる。

【0032】請求項9記載の発明では、請求項1～請求項3または請求項5記載のプログラム動作検証システムで、前記した所定の装置は通信ケーブルによって接続された複数の装置によって構成されており、それぞれの装置に対して被測定プログラムのコンポーネントが割り振られており、表示手段はそれぞれの装置単位で該当するコンポーネントごとの表示を行うことを特徴としている。

【0033】すなわち請求項9記載の発明では、1つの被測定プログラムが複数の装置によって実行されるとき、それぞれの装置におけるコンポーネントごとの検証結果は装置ごとに表示される例を示している。これにより、各装置側で自己に関係するコンポーネントの不具合に対するハードウェア側の原因を探る等の対処を容易に行うことができる。

【0034】

【発明の実施の形態】

【0035】

【実施例】以下実施例につき本発明を詳細に説明する。

【0036】図1は本発明の一実施例におけるプログラム動作検証システムの構成の概要を示したものである。本実施例のシステムは、被測定計算機101と測定装置102とによって構成されている。被測定計算機101は、通常のパーソナルコンピュータと同一の構成となっており、CPU（中央処理装置）111およびハードディスク、RAM（ランダム・アクセス・メモリ）等からなるメモリ112の他に、測定装置102にデータを送力するためのデータ出力部113を備えており、それらがバス114に接続されている。メモリ112内にはこの被測定計算機101の基本的な制御を行わせるためのプログラムとしてのOS115とこのOS115上で実行される被測定プログラム116が格納されている。被

測定プログラム116には、プログラム動作記述を基にして測定点が埋め込まれている。

【0037】なお、この図ではキーボード等の入力機器（図示せず）やディスク読取装置等の装置からデータの送力を行うデータ入力部やディスプレイ（図示せず）の表示制御を行う表示制御部等の回路装置は本発明の動作と直接関係しない。そこで、これらの図示を省略している。

【0038】被測定計算機101のOS115の基本機能を実現する部分としてのカーネル内には、CPUや磁気ディスク、光ディスクといった計算機資源の使用状況を示すイベントを送力するためにカーネルブローブ117が埋め込まれている。被測定プログラム116が実行され、計算機資源の使用状況のうちの特定のものがカーネルブローブ117によって検出されると、これらの使用状況がイベントデータ118として測定装置102に送力される。このカーネルブローブ117の他にプログラム中に埋め込まれた測定点を検出するブローブ（以下、測定点ブローブという。）も存在している。測定点ブローブもこれが実行されたときイベントデータ118を送力出力線119を介して測定装置102に送力するようにになっている。

【0039】一方、測定装置102はデータ出力線119によって受信した被測定計算機からのイベントデータ118を記録装置121に送力するようにになっている。記録装置121には測定装置102内に設けられた時計回路122から時刻データ123も供給されるようになっている。記録装置121はイベントデータ118と時刻データ123を対応付けて測定データ124として記録する。この測定データ124を基にして、ある測定点から他の測定点に至るまでの経過時間が検証したい動作条件としての時間範囲（設計値）に合致しているかどうかの判別が行われる。これについては次に説明する。

【0040】図2は、本実施例のプログラム動作検証システムの動作の概要を示したものである。図1に示した被測定プログラム116を作成するために、プログラムの作成時にプログラムは性能設計情報131を含んだプログラム動作記述132を作成する。プログラム動作記述132には、予めプログラムがプログラムの動作を実際の装置で検証するための測定の開始点および終了点を示す位置を記している。プログラムの実行に関する性能条件を表わした性能設計情報131として、本実施例では測定点による測定の開始から終了までに許容できる所要時間としての設計値が記述されている。

【0041】測定点抽出手段133は、各コンポーネントの境界位置のような測定点をプログラム動作記述132から抽出し、これを所定の記憶領域に測定点情報135として格納する。これらの測定点情報135は、プログラムコード136の該当する位置に測定点埋込手段137によって埋め込まれる。被測定計算機101内で測



定点埋込手段137はOS115によって所定の制御を行うCPU111が該当する。このような埋め込み操作によって被測定プログラム116 (図1参照) が作成される。

【0042】被測定プログラム116は測定点から他の測定点まで所要時間を測定する装置 (たとえば図1に示した測定装置102) からなる測定手段139によって測定され、その結果としての測定データ124が作成される。測定データ124は、データ分析比較手段142に10 入力される。図1ではこれを記録装置121が兼用している。データ分析比較手段142は、プログラム動作記述132内の性能設計情報131から測定点の開始から終了までに予想される時間の許容値 (設計値) と実際に費やした時間を比較して、被測定プログラム116の動作が正常であるかどうかの検証結果を図1に示す記録装置121のような装置でプリントアウトしたり、図示しないディスプレイに表示するようになっている。

【0043】図3は、本実施例のプログラム検証用のクライアントとサーバの関係を示したものである。クライアント161はインターネットで通信を行うための通信ケーブル162を介してサーバ163と接続されている。なお、ここでは説明を簡単にするためにクライアント161とサーバ163を1対1の関係で図示している。図1に示した被測定計算機101と測定装置102は、図示していないが通信ケーブルでこれらクライアント161およびサーバ163と接続されている。

【0044】図4は、プログラムによるプログラム動作記述の例を示したものである。本実施例ではUML (Unified Modeling Language) を使用して記述を行っている。図で左側がクライアント161側の記述であり、右側がサーバ163側の記述である。これらの記述で、枠で示した“{”と“}”で囲まれた箇所が性能設計情報171~173である。

【0045】この言語の記述を簡単に説明すると、“Begin Client Process”という記述によってクライアント161が処理を開始し、“Send Request To Server”という記述でたとえばある処理をサーバ163側に要求している。この“Send Request To Server”という処理の要求に付帯して、以下のような性能設計情報171が記述されている。

{Send Request To Server - Begin Client Process < 1 sec} (性能設計情報171)

【0046】この性能設計情報171は、不等式の左辺で示した2点間、すなわち“Begin Client Process”から“Send Request To Server”までの間に消費するクライアント161側のCPUの経過時間が右辺に示した“1 sec” (1秒) よりも短いという条件を示している。この条件は、プログラムを処理する装置側等に何らかの不具合がなければ、達成される範囲の値である前記した設計値として記載される。したがって、この場合には、

“Begin Client Process”と“Send Request To Server”の記述が行われたプログラムの箇所がそれぞれ測定点181、182となる。

【0047】“Send Request To Server”という処理の要求の記述に基づいて、クライアント161側からサーバ163側にリクエスト (Request) が行われる。サーバ163では、“Receive Request From Client”という記述の箇所でクライアント161から処理の要求を受け、“Send Reply To Client”という記述の箇所で返答 (Reply) をクライアント161に返す。サーバ163側におけるこの箇所には以下のような性能設計情報172が記述されている。

{Send Reply To Client - Receive Request From Client < 3 sec} (性能設計情報172)

【0048】この性能設計情報172は、不等式の左辺で示した2点間、すなわち“Receive Request From Client”から“Send Reply To Client”までに消費するサーバ163側の経過時間が、右辺に示した“3 sec” (3秒) よりも短いという条件を示している。この条件も、プログラムを処理する装置側等に何らかの不具合がなければ、達成される範囲の値である前記した設計値として記載される。したがって、この場合には、“Receive Request From Client”と“Send Reply To Client”の記述が行われたプログラムの箇所がそれぞれ測定点183、184となる。

【0049】クライアント161は“Receive Reply From Server”という記述の箇所でサーバ163側の返答 (Reply) を受ける。そしてそれから所定の作業を行った後に“End Client Process”で所定の処理を完了させる。この記述の箇所には以下のような性能設計情報173が記述されている。

{End Client Process - Receive Reply From Server < 2 sec} (性能設計情報173)

【0050】この性能設計情報173は、不等式の左辺で示した2点間、すなわち“End Client Process”から“Receive Reply From Server”までに消費するクライアント161の経過時間が右辺に示した“2sec” (2秒) よりも短いという条件を示している。この条件も、プログラムを処理する装置側等に何らかの不具合がなければ、達成される範囲の値である前記した設計値として記載される。したがって、この場合には、“Receive Reply From Server”と“End Client Process”の記述が行われたプログラムの箇所がそれぞれ測定点185、186となる。

【0051】図1に示したCPU111はOS115によって、このようなUMLによる記述を解釈する。そして、クライアント161側であれば、対応するそれぞれの測定点181、182、185、186と性能設計情報171、173をそのメモリ112に一時的に格納し、これを基にしてプログラム動作の検証を行うことに

なる。サーバ163側も同様の制御を行うがその結果を独自にサーバ163側で表示することもできるし、クライアント161側に返すような処理形態を採ることも可能である。

【0052】図5は、クライアント側のプログラムコードとしてのクライアントプログラムの流れの要部を示したものである。まず、図4の測定点181に対応するプリプロセス (Pre Process) (ステップS201)から測定点182に対応するサーバ163側への処理要求 (Send Request To Server) (ステップS202)へ移行し、更にサーバ163側からの返答 (Receive Reply From Server) (ステップS203)を受信し、所定のプロセスを終了させる (Post Process) (ステップS204)ことになる。

【0053】図6は、サーバ側のプログラムコードとしてのサーバプログラムの流れの要部を示したものである。サーバは、クライアント161から処理の要求を受ける (Receive Request From Client) (ステップS221)と、その要求された処理を実行し (Process Request) (ステップS222)、クライアント161に返答する (Send Reply To Client) (ステップS223)という処理を繰り返す。したがって、サーバ163のプログラムには実行の終了点は存在しない。

【0054】次に図2に示した測定点抽出手段133による測定点の抽出動作を具体的に説明する。図1に示したCPU111とOS115によって実現する測定点抽出手段133は、プログラム動作記述内の性能設計情報171~173を用いてそれぞれの測定点181~186を抽出するようになっている。図4に示した例の場合には、性能設計情報171~173として記述されている不等式の左辺で引用されている点、すなわち“Begin Client Process”、“Send Request To Server”、“Receive Request From Client”、“Send Reply To Client”、“Receive Reply From Server”および“End Client Process”の6点が測定点181~186を表わした測定点情報として抽出される。

【0055】図7は、これら抽出された測定点情報を埋め込んで作成した被測定プログラムを用いたクライアント側の処理を示したものであり、図8はイベントを出力するそれぞれのプローブとソフトウェアイベントを対応させて示したものである。図7の処理は図5の処理と対応している。クライアント161は、前記したプログラム中に埋め込まれた測定点を検出する測定点プローブの1つとしてのプローブ“Prb Cb”によってプリプロセス (Pre Process) を検出し (ステップS241)、クライアント処理を開始する。このとき、プローブ“Prb Cb”によってクライアント処理が開始した旨のイベントデータがデータ出力部113 (図1)を介して被測定計算機101から測定装置102に送られる。測定装置102では、このイベントデータおよび時計回路122の

出力する時刻データ123を記録装置121に送り、イベントデータごとに時刻を記録して測定データ124を作成していく。

【0056】さて、図7のステップS241で測定点のうちの開始点が検出されたら、次の測定点プローブ“Prb Cs”によってサーバ163への処理要求 (Send Request To Server) が検出されて (ステップS242)、その旨のイベントデータがデータ出力部113から測定装置102に送られる。この後、ステップS243でサーバ163側から返答を受ける (Receive Reply From Server) まで、クライアント161はスリープ (Sleep) 状態となる。これはカーネルプローブ117で検出されて、同様にそのイベントが記録装置121側に送られて記録される。また、サーバ163側はクライアント161からの処理要求 (Send Request To Server) によってウエイクアップ (Wake Up) するが、同様にこれはカーネルプローブ117で検出されて、記録装置121側に送られて記録されることになる。

【0057】ステップS243でサーバ163側からの返答 (Receive Reply From Server) があると、測定点プローブの1つとしてのプローブ“Prb Cr”がこれを検出して、同様に図8に示すようなイベントデータを出力する。そしてその後、サーバ163の返答に基づいた所定のプロセスを終了させる (Post Process) (ステップS204)と、測定点プローブの1つとしてのプローブ“Prb Ce”がこれを検出して、同様に図8に示すようなイベントデータを出力することになる。

【0058】図9は、被測定プログラムを用いたサーバ側の処理を示したものであり、図6の処理と対応している。ステップS261ではクライアント161から要求を受信 (Receive Request From Client) し、これが測定点プローブの1つとしてのプローブ“Prb Sr”で検出される。プローブ“Prb Sr”は図8に示したように、サーバ163がクライアント161から処理要求を受信した旨のイベントデータを出力する。ステップS262ではサーバ163が要求された処理を実行 (Process Request) する。処理が終了すると、測定点プローブの1つとしてのプローブ“Prb Ss”がこれを検出して、クライアントに返答する (Send Reply To Client) (ステップS263)。この後、サーバがスリープ (Sleep) 状態となり、クライアント161側がウエイクアップ (Wake Up) する。これらはカーネルプローブ117で検出されて、記録装置121側に送られて記録されることになる。それぞれのプローブによってイベントデータが出力され、記録される。

【0059】ところで、図2に示したデータ分析比較手段142は、本実施例ではクライアント161に設けられている。すなわち、クライアント161側で発生したイベントデータだけでなく、サーバ163側で発生したイベントデータもクライアント161側に集中的に集め

られて、性能設計情報 131 との比較が行われる。データ分析比較手段 142 は、測定の開始点から終了点までの経過時間が性能設計情報 131 として記述された条件を満足する場合には“Good”（条件満足）という結果を実際に要した経過時間と共に表示し、条件を満足しなかった場合には“Not Good”（条件不満足）という結果を実際に要した経過時間と共に表示する。

【0060】図 10 は、このプログラム動作検証システムで記録装置あるいは図示しないディスプレイが出力あるいは表示する検証結果の一例を示したものであり、図 4 に対応させたものである。実際の経過時間としての実行値と設計値および条件を満たしたかどうかの結果とが、クライアント 161 とサーバ 163 の関係で一目でわかるように表示されている。たとえばプログラムあるいは装置の設計者は、サーバ 163 がクライアント 161 から処理の要求を受けてから返答を返すまでの時間が設計値である 3 秒よりも長い 4.2 秒かかったことを直ちに知ることができる。すなわち、プログラムコードを実行させたときの“Receive Request From Client”と“Send Reply To Client”（図 4 参照）の間の実行性能が性能設計情報 172 で示されている性能条件を満足していないことを知り、必要な対策を採ることができる。

#### 【0061】第 1 の変形例

【0062】図 11 は先の実施例の図 1 に対応するもので被測定計算機の変形例を示したものである。図 11 で図 1 と同一部分には同一の符号を付しており、これらの説明を適宜省略する。先の実施例では被測定計算機 101 と測定装置 102 は別々の装置として構成されていたが、これらを同一の装置で構成することができる。たとえばパーソナルコンピュータとこれに接続された記録装置あるいはディスプレイといったような構成を挙げることができる。この第 1 の変形例の被測定計算機 301 では、CPU 302 内の時計回路 303 を時刻データ 123 の出力用に使用している。また、この時刻データ 123 や OS 115 から出力されるイベントデータ 118 等のデータは測定データ 124 としてメモリ 112 内に格納される。これらのデータが性能設計情報 131 と比較され、その結果が図示しないプリンタでプリントアウトされたり、同じく図示しないディスプレイに視覚的に表示されることになる。

【0063】この第 1 の変形例では、被測定計算機 301 を実施例のクライアントと同一のものとして構成することができ、この場合には測定および検証を行うための特別なハードウェアを必要としないという利点が生じる。もちろん、実施例でもその被測定計算機 101 をクライアント 161 と同一のものとして構成することができる。

#### 【0064】第 2 の変形例

【0065】図 12 は、本発明の第 2 の変形例におけるプログラム動作検証システムの構成を示したものであ

る。このシステムはネットワーク 401 によって接続されたクライアント 402 とサーバ 403 によって構成されている。クライアント 402 およびサーバ 403 はそれぞれ図 1 に示したものと基本的に同一構成の被測定計算機 101C、101S と測定装置 102C、102S によって構成されている。そこで、これらの構成部品で図 1 に示したものと同一のものには同一の数字とアルファベット C または S を付しており、これらの部品の説明を適宜省略する。測定装置 102C 内の時計回路 122C と測定装置 102S 内の時計回路 122S とは時刻同期用通信ケーブル 404 で接続されており、両時計回路 122C、122S の出力する時刻データ 123C、123S の同期がとられている。

【0066】クライアント 402 の被測定計算機 101C 上ではクライアントプログラムが動作し、サーバ 403 の被測定計算機 101S 上ではサーバプログラムが動作するようになっている。クライアント 402 およびサーバ 403 はそれぞれ単独に測定装置 102C あるいは 102S を配置しているので、それぞれ自分の側でイベントデータを時刻データと共に記録し、性能設計情報と比較することができるようになっている。もちろん、ネットワーク 401 で接続先にこれらの比較結果を送ることで、ネットワーク全体のどの装置で処理が滞っているといった情報も取得し、それぞれの計算機で表示したりプリントアウトすることもできる。

【0067】図 13 は、この第 2 の変形例でクライアントがサーバに所定の処理要求を行って処理結果を得た場合のそれぞれの被測定計算機のソフトウェアイベントとプローブの関係を示したものである。この例は、処理内容的には実施例の図 8 で説明したクライアント 161 とサーバ 163 の処理と同一であるので、詳細な説明は省略する。先の実施例では、測定装置がシステムに 1 つだけだったので、図 10 で示したような比較結果が 1 箇所

で集中的に行われるが、第 2 の変形例では自分の装置に関係する分だけを部分的に表示したり、必要によりシステム全体の比較結果を表示することもできる。

【0068】なお、実施例および第 2 の変形例では 1 つのクライアントと 1 つのサーバの関係を説明したが、1 つの計算機から複数の計算機に順次処理が移行していくものや、1 つの処理を複数の計算機が分散して行うような場合にも本発明を同様に適用することができる。

【0069】更に実施例では図 10 に示すように、実際の経過時間としての実行値と設計値および条件を満たしたかどうかの結果とがディスプレイに表示されたりプリンタでプリントアウトされるようにしていた。これとは異なり、それぞれの処理に対しての実行値のみがコンポーネントごとに表示されるものであってもよいし、設計値との比較のみが表示されるものであってもよい。

【0070】また実施例および変形例では測定点間の経過時間を性能設計情報に記述しその結果を表示させる場

10

20

30

40

50

合について説明したが、これ以外の性能設計情報、たとえばCPUの稼働率等を併せてこれらの測定結果を合わせて表示するようにしてもよい。

#### 【0071】

【発明の効果】以上説明したように請求項1および請求項4記載の発明によれば、プログラム動作記述の性能設計情報に記述した内容を基にして測定開始点と測定終了点を抽出するので、たとえばプログラマが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所10で測定を開始したり終了させることができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージの送受信点といったように予め決まったルールを使用したり、OS自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での処理時刻を求め、これらの差から経過時間を算出して表示するので、実際にかかった時間を被測定プログラムの対応する箇所に対応付けて簡単にチェックすることができる。

【0072】また請求項2および請求項4記載の発明によれば、プログラム動作記述の性能設計情報に記述した内容を基にして測定開始点と測定終了点を抽出するので、たとえばプログラマが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所20で測定を開始したり終了させることができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージの送受信点といったように予め決まったルールを使用したり、OS自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での処理時刻を求め、これらの差から経過時間を算出して表示するので、実際にかかった時間を被測定プログラムの対応する箇所に対応付けて簡単にチェックすることができる。更に本発明では、算出した経過時間を設計値と共に被測定プログラムの対応する箇所に表示するので、算出した経過時間が設計値を満足しているか否かを簡単にチェックすることができる。

【0073】更に請求項3および請求項4記載の発明によれば、プログラム動作記述の性能設計情報に記述した内容を基にして測定開始点と測定終了点を抽出するので、たとえばプログラマが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所40で測定を開始したり終了させることができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージ

の送受信点といったように予め決まったルールを使用したり、OS自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での処理時刻を求め、これらの差から経過時間を算出しこれを設計値と比較した判別結果を被測定プログラムの対応する箇所に対応付けて表示するので、チェックするコンポーネントが多いような場合にもそれぞれの所要時間が設計値を満足しているかどうかを短時間にチェックすることができる。

【0074】また請求項5記載の発明によれば、プログラム動作記述の性能設計情報に記述した内容を基にして測定開始点と測定終了点を抽出するので、たとえばプログラマが性能設計情報として測定の開始点や終了点を判読できるデータを書き込んでおくことにより、被測定プログラムの所望の箇所50で測定を開始したり終了させることができる。したがって、プログラムにおける測定点の抽出をソフトウェア技術者あるいは測定実施者が手作業で行う必要がない。また、メッセージの送受信点といったように予め決まったルールを使用したり、OS自体が用意している機能を利用する場合と異なり、被測定プログラムの所望のコンポーネントの開始点と終了点を任意に指定して測定を行うことができる。そしてこれらの点での時刻を求め、これらの差から経過時間を算出しこれを設計値と比較した判別結果を被測定プログラムの対応する箇所に対応付けて表示するので、チェックするコンポーネントが多いような場合にもそれぞれの経過時間が設計値を満足しているかどうかを短時間にチェックすることができる。

【0075】更に請求項6記載の発明によれば、請求項5記載の発明による表示内容に加えて経過時間算出手段によって算出した実際の経過時間も表示するようにしているので、この経過時間を設計値と対比することでプログラムの実行について何らかの不具合があったときにその原因を類推することができる。

【0076】また請求項7記載の発明によれば、1つの被測定プログラムが複数の装置によってコンポーネント単位で実行される場合に、コンポーネントごとの経過時間等の性能指標値を表示したりそれぞれの設計値と比較することが可能になる。

【0077】更に請求項8記載の発明によれば、1つの被測定プログラムが複数の装置によって実行される場合でも、それぞれの装置におけるコンポーネントごとの検証結果を集中的に調べることができ、プログラムの全体的な動作の把握を短時間に行うことができる。

【0078】また請求項9記載の発明によれば、1つの被測定プログラムが複数の装置によって実行されるとき、それぞれの装置におけるコンポーネントごとの検証結果を担当する装置ごとに表示するので、これらの装置ごとにハードウェア側の原因を探る等の対処を容易に行

うことができる。

【図面の簡単な説明】

【図 1】本発明の一実施例におけるプログラム動作検証システムの構成の概要を示したブロック図である。

【図 2】本実施例のプログラム動作検証システムの動作の概要を示した説明図である。

【図 3】本実施例のプログラム検証用のクライアントとサーバの関係を示したブロック図である。

【図 4】本実施例でプログラマによるプログラム動作記述の例を示した説明図である。

【図 5】本実施例でクライアント側のプログラムコードとしてのクライアントプログラムの流れの要部を示した流れ図である。

【図 6】本実施例でサーバ側のプログラムコードとしてのサーバプログラムの流れの要部を示した流れ図である。

【図 7】本実施例で測定点情報を埋め込んで作成した被測定プログラムを用いたクライアント側の処理を示した説明図である。

【図 8】本実施例でイベントを出力するそれぞれのプローブとソフトウェアイベントを対応させて示した説明図である。

【図 9】本実施例で被測定プログラムを用いたサーバ側の処理を示した説明図である。

【図 10】本実施例のプログラム動作検証システムで記録装置あるいは図示しないディスプレイが出力あるいは表示する検証結果の一例を示した平面図である。

【図 11】本発明の第 1 の変形例として被測定計算機の

構成を示したブロック図である。

【図 12】本発明の第 2 の変形例におけるプログラム動作検証システムの構成を示したブロック図である。

【図 13】第 2 の変形例でクライアントがサーバに所定の処理要求を行って処理結果を得た場合のそれぞれの被測定計算機のソフトウェアイベントとプローブの関係を示す説明図である。

【符号の説明】

101、301 被測定計算機

102 測定装置

111 CPU (中央処理装置)

112 メモリ

115 OS

116 被測定プログラム

117 カーネルプローブ

121 記録装置

122、303 時計回路

124 測定データ

132 プログラム動作記述

133 測定点抽出手段

137 測定点埋込手段

139 測定手段

142 データ分析比較手段

161、402 クライアント

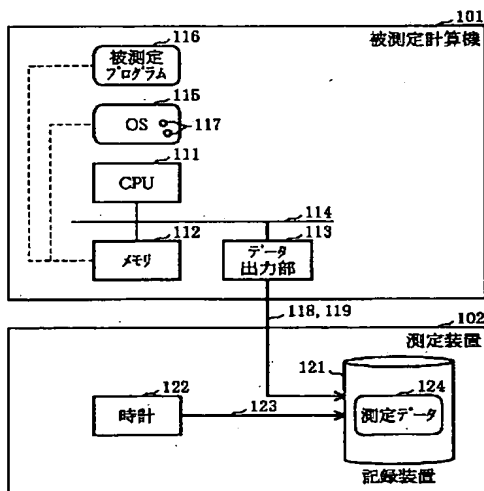
163、403 サーバ

171~173 性能設計情報

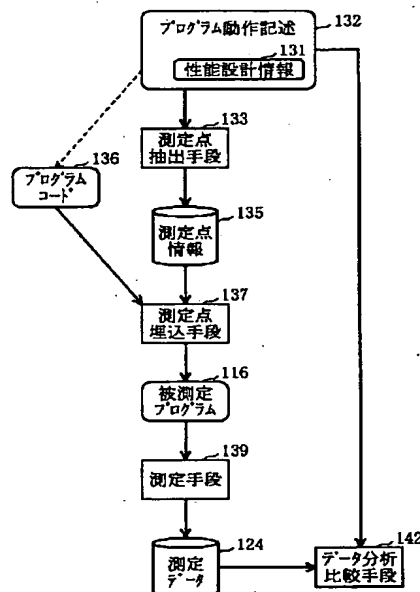
181~184 測定点

401 ネットワーク

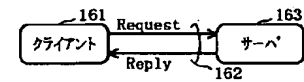
【図 1】



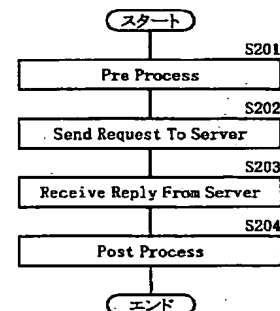
【図 2】



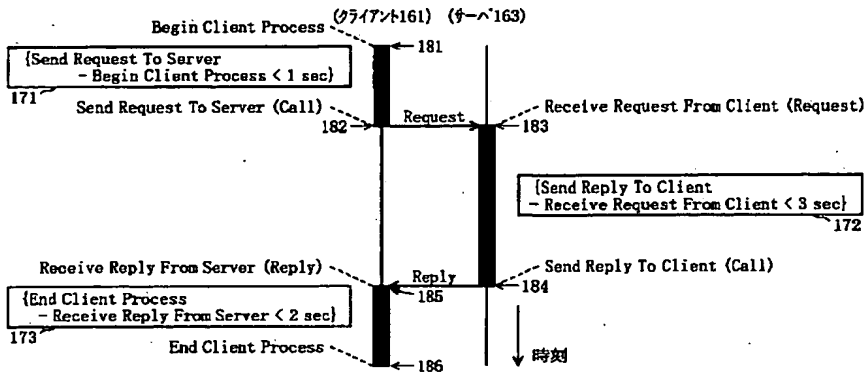
【図 3】



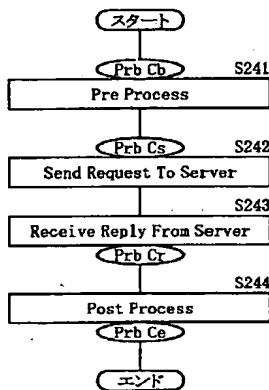
【図 5】



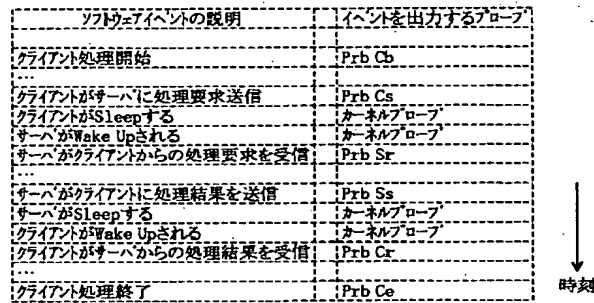
【図4】



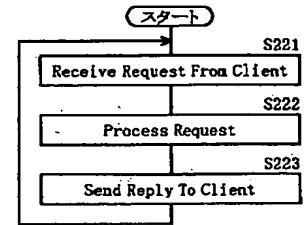
【図7】



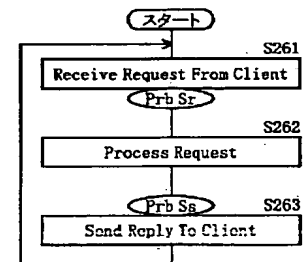
【図8】



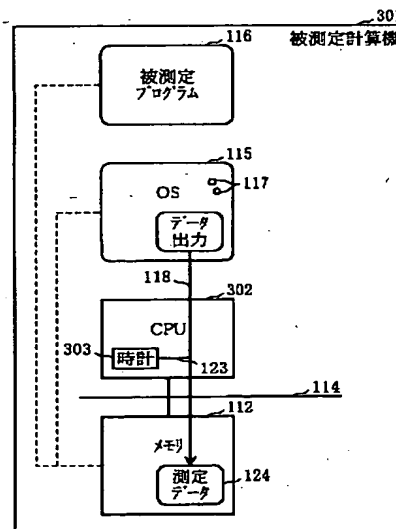
【図6】



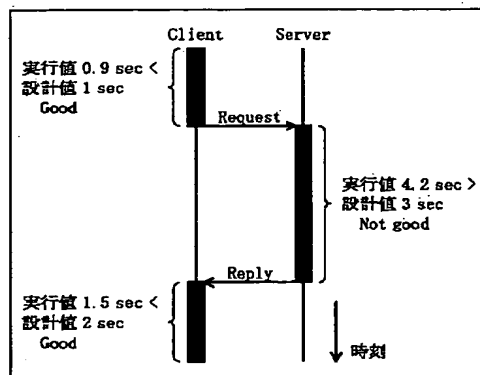
【図9】



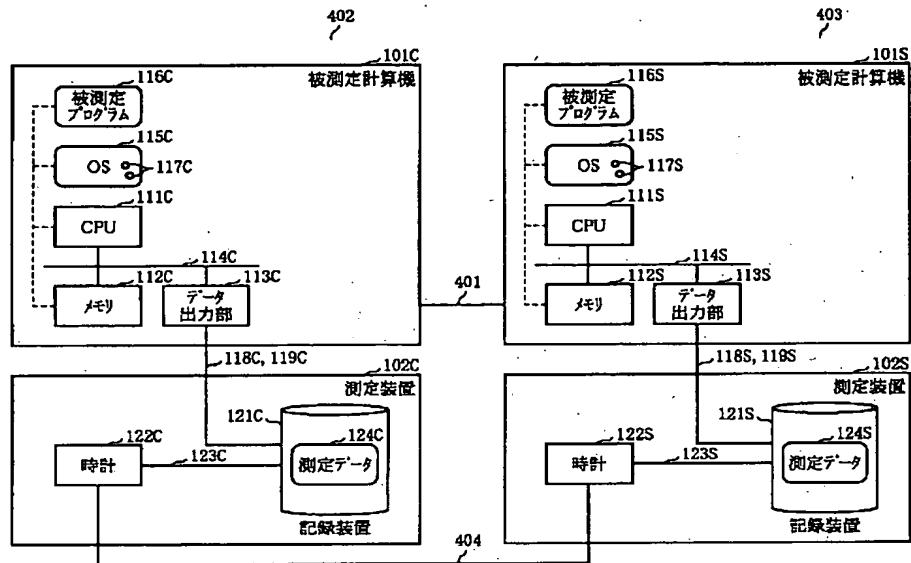
【図11】



【図10】



【図12】



【図13】

